ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

Laboratorio di
Sicurezza Informatica

Esercitazione: VPN

Andrea Melis
Marco Prandini
Dipartimento di Informatica – Scienza e Ingegneria

Agenda

- Creare un test-bed per sperimentare topologie di connessioni differenti
  - Site-to-site
  - Host-to -site
- Configurare una connessione VPN tra due server di subnet distinte passando per un tunnel
  - creato con IPSec
  - creato con OpenVPN
  - troubleshooting
- Esercitazioni proposte: sprimentare la modalità transport
  - con IPSec
  - con OpenVPN

Test-bed

- Queste operazioni preliminari ci consentiranno di creare un'architettura di rete utilizzabile per tutti i test
- L'architettura crea delle macchine virtuali headless e utilizza le seguenti tecnologie:
  - vagrant con virtualbox
  - ansible
- Per approfondimenti sull'utilizzo dei tool
  - https://www.vagrantup.com/
  - https://www.ansible.com/

Creazione dei test-bed

- Nel PC Host, scaricate da Virtuale il file **di configurazione dell'infrastruttura secnet.zip ed estratelo**
- Una volta estratta la cartella conterrà:
  - File secnet.sh
  - File vagrant.tgz
- A seconda che lavoriate sul PC da casa o successivamente dal PC del laboratorio lanciate:
  - Dal PC da casa estrate l'archivio vagrant.tgz con
    - tar -xf vagrant.tgz
    - cd vagrant
    - vagrant up
  - Dal PC del laboratorio è sufficente lanciare
    - ./secnet.sh

File VPN

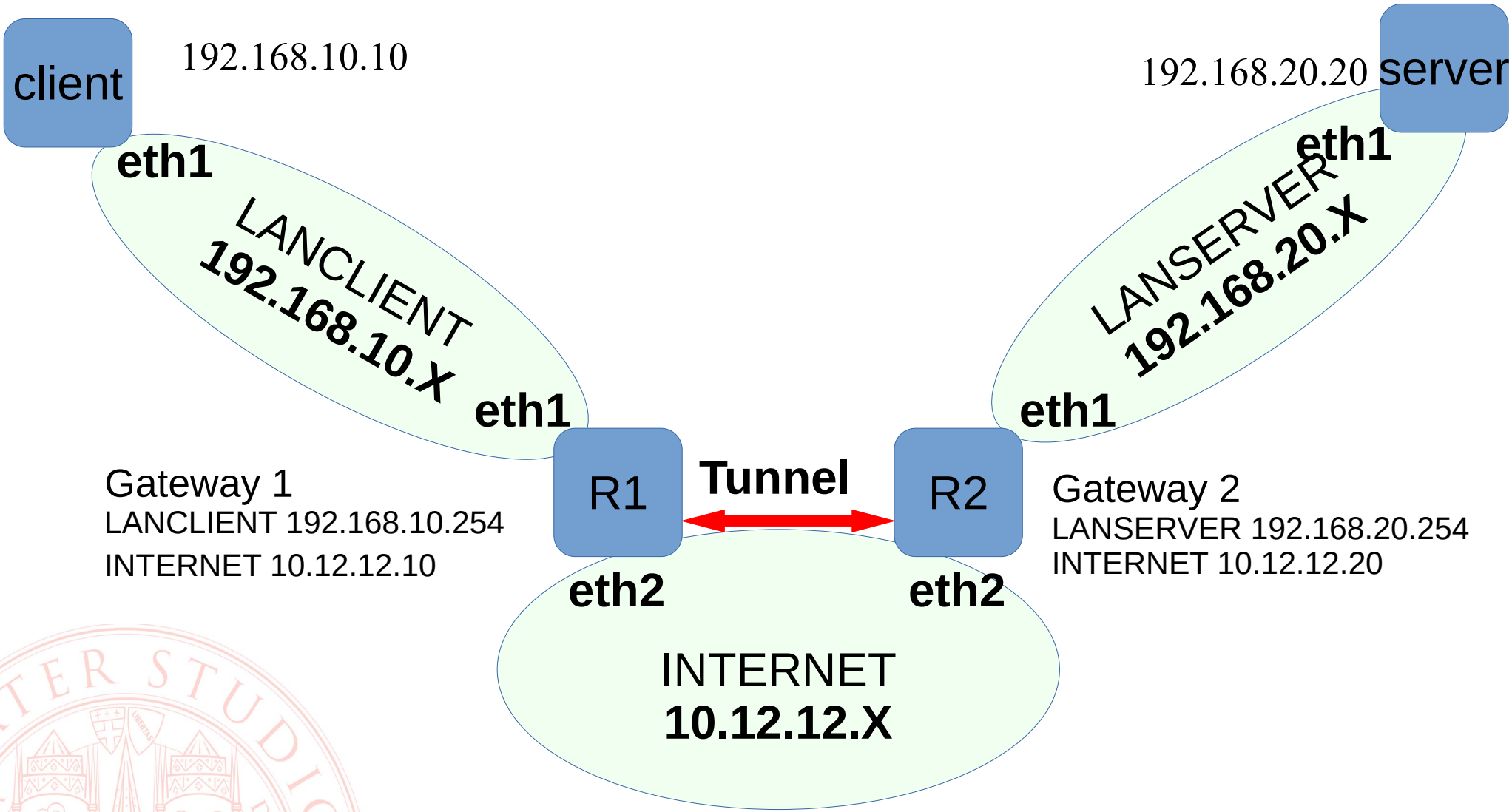
- Una volta che l'infrastruttura è stata creata correttamente dovremmo caricare i file di configurazione delle vpn sull VM.
- Prima di tutto scarichiamo il file vpn_files.zip e decomprimiamolo
- L'archivio contiene 4 cartelle, una per ogni esercitazione. Dobbiamo quindi caricarli sulle VM. Questo passaggio è possibile eseguirlo in due modi.
- Modalità Lazy: Si possono copiare e incollare i file sul terminale attraverso nano / vi
- Modalità Smart: Vagrant per ogni infrastruttura deployata crea una cartella condivisa nella cartella vagrant stessa da dove lanciate il comando vagrant up

Cartella condivisa

- Modalità Smart: Vagrant per ogni infrastruttura deployata crea una cartella condivisa nella cartella vagrant stessa da dove lanciate il comando **vagrant up**

- All'interno di questa cartella sarà quindi possibile inserire tutti i file necessari, che saranno poi visualizzabili all'interno delle VM stesse nel percorso **/vagrant**

Topologia


client
192.168.10.10
eth1
LANCLIENT
192.168.10.X
eth1
192.168.20.20
server
eth1
LANSERVER
192.168.20.X
eth1
Gateway 1
LANCLIENT 192.168.10.254
INTERNET 10.12.12.10
R1
Tunnel
R2
Gateway 2
LANSERVER 192.168.20.254
INTERNET 10.12.12.20
eth2
eth2
INTERNET
10.12.12.X

Verifichiamo la configurazione

- Utilizziamo ilcomando
  - `ping DESTINAZIONE`
    - deve funzionare da client a R1 e viceversa
      - `(da host) ping 192.168.10.254`
      - `(da R1) ping 192.168.10.10`
    - deve funzionare da server a R2 e viceversa
      - `(da host)ping 192.168.20.254`
      - `(da R2) ping 192.168.20.20`
    - deve funzionare da R1 a R2 e viceversa
      - `(da R1) ping 10.12.12.20`
      - `(da R2) ping 10.12.12.10`
    - non deve funzionare per le altre coppie possibili
      - `(da host) ping 192.168.20.20`
      - `(da R1) ping 192.168.20.20`

IPSec

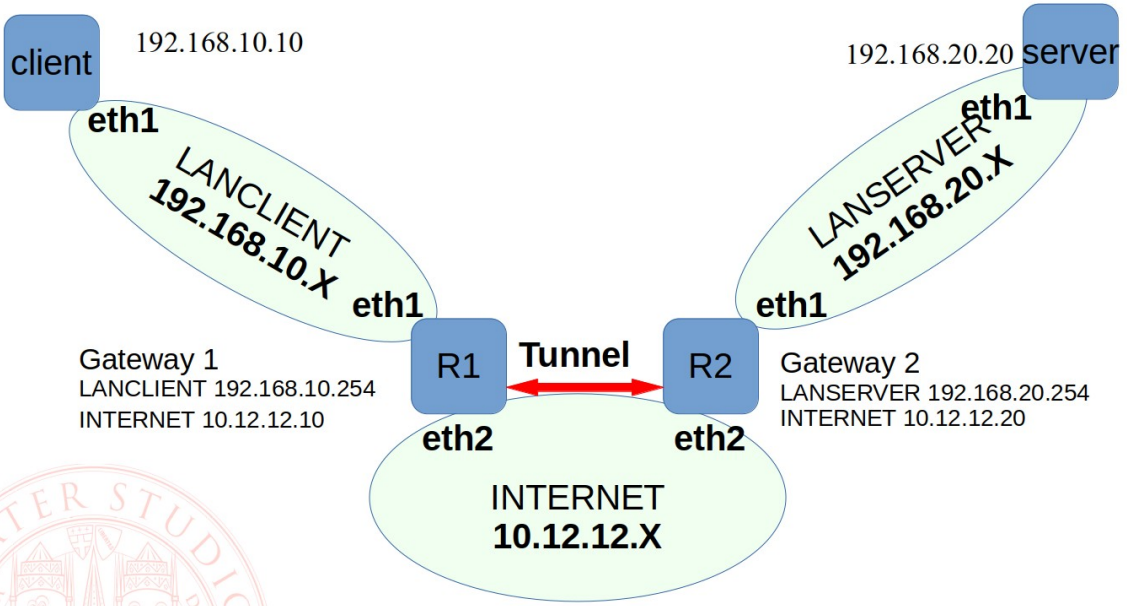
- A questo punto possiamo passare alla configurazione del tunnel di IPSec
- I file da aggiungere/modificare sono 2:
  - /etc/ipsec.conf
    - contiene tutte le configurazioni del tunnel e delle sottoreti che il tunnel ipsec “vede”
  - /etc/ipsec.secrets
    - contiene le credenziali dei nodi del tunnel ipsec.

Commenti alla configurazione

- Vediamo cosa significano le direttive contenute nei file che abbiamo installato
- L'implementazione che useremo (Strongswan) identifica i due lati del tunnel come *left* e *right*
- Non corrispondono necessariamente alla stessa visione, cioè i due lati possono usare gli stessi termini per riferirsi agli stessi elementi, oppure invertirli
  - è l'uso più comune:
  - left = lato locale
  - right = lato remoto

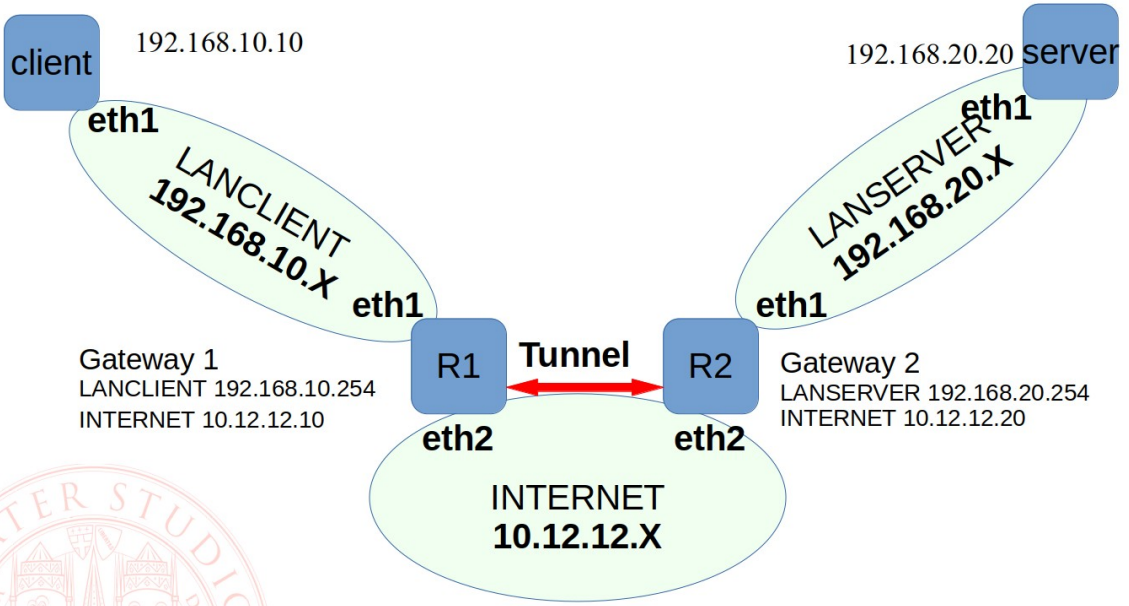
Configurazione di R1 – ipsec.conf

```
config setup

conn vpn1
authby=secret
auto=start
compress=no
pfs=yes
type=tunnel
    left=10.12.12.10
    leftsubnet=192.168.10.0/24
    right=10.12.12.20
    rightsubnet=192.168.20.0/24

include /var/lib/strongswan/ipsec.conf.inc
```



Configurazione di R2 – ipsec.conf

```
config setup

conn vpn1
authby=secret
auto=start
compress=no
pfs=yes
type=tunnel
    left=10.12.12.20
    leftsubnet=192.168.20.0/24
    right=10.12.12.10
    rightsubnet=192.168.10.0/24

include /var/lib/strongswan/ipsec.conf.inc
```



Configurazione dell'autenticazione

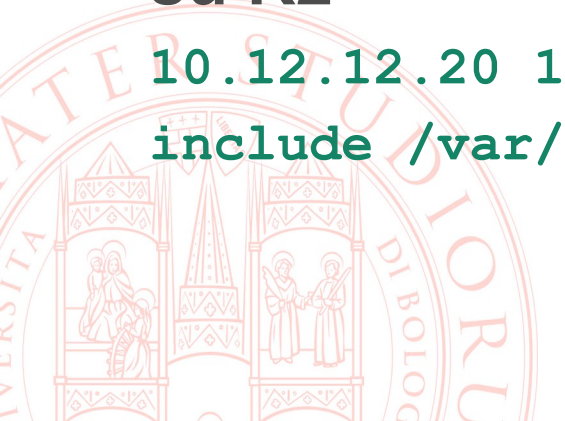
- IPSec può utilizzare una varietà di meccanismi anche molto complessi di verifica dell'identità dei nodi
- La direttiva **authby=secret** imposta il meccanismo più semplice: una **P**re-**S**hared **K**ey memorizzata di default nel file **ipsec.secrets**

- su R1

```
10.12.12.10 10.12.12.20 : PSK "password"
include /var/lib/strongswan/ipsec.secrets.inc
```

- su R2

```
10.12.12.20 10.12.12.10 : PSK "password"
include /var/lib/strongswan/ipsec.secrets.inc
```



Configurazione di Ipsec Completata

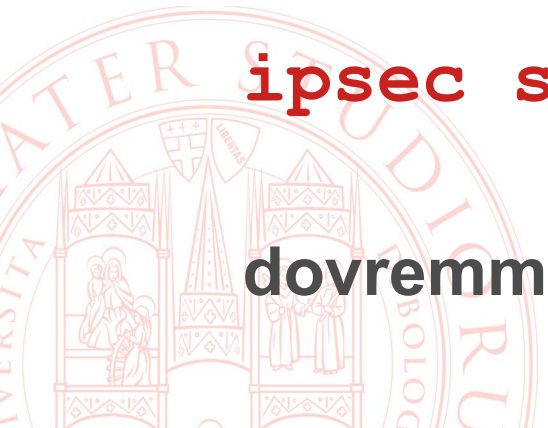
- A questo punto riavviamo il demone di ipsec e dovremmo vedere lo status attivo

```
ipsec stop
systemctl restart ipsec.service
systemctl status ipsec.service
```

- Attendiamo qualche secondo e con il comando

```
ipsec status
```

dovremmo poter vedere il tunnel attivo!



Configurazione di Ipsec Completata

```
ipsec status
Security Associations (1 up, 0 connecting):
        vpn1[6]: ESTABLISHED 15 minutes ago,
10.12.12.10[10.12.12.10]...10.12.12.20[10.12.12.20]
        vpn1{18}:  INSTALLED, TUNNEL, reqid 5, ESP SPIs:
c3fb9d79_i c52dfe88_o
        vpn1{18}:   192.168.10.0/24 === 192.168.20.0/24
```

Troubleshooting e Comandi Utili

- Comando più verboso di ipsec dove è possibile vedere il numero esatto di pacchetti in entrata e uscita

```
ipsec statusall
...descrizione tunnel tra cui
penultima riga
252 bytes_i (3 pkts, 3s ago), 252
bytes_o (3 pkts, 3s ago)
pacchetti in entrate e in uscita
```

Troubleshooting e Comandi Utili

- Vedere eventuali errori guardate i log di syslog

```
cat /var/log/syslog
```

...possibile errore

```
received INTERNAL_ADDRESS_FAILURE
notify, no CHILD_SA built
```

significa che il tunnel è attivo ma le sottoreti figlie non sono state riconosciute.

Troubleshooting e Comandi Utili

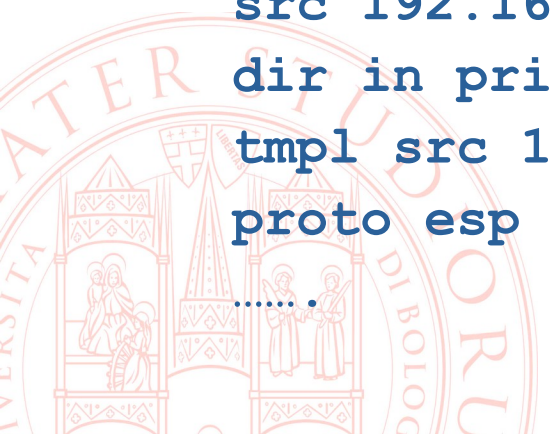
- Tutti i comandi **ip xfrm**
- **xfrm** è un framework per “trasformare” pacchetti, come ad esempio crittarne il payload, usato quindi per ipsec:
- ip-xfrm - transform configuration

```
ip xfrm state
...stato del tunnel
src 10.12.12.20 dst 10.12.12.10
…….
src 10.12.12.10 dst 10.12.12.20
…….
```

Troubleshooting e Comandi Utili

- Vedere tutte le policy ipsec

```
ip xfrm policy
src 192.168.20.0/24 dst 192.168.10.0/24
dir out priority 375423 ptype main
tmpl src 10.12.12.20 dst 10.12.12.10
proto esp spi 0xce5de214 reqid 1 mode tunnel
src 192.168.10.0/24 dst 192.168.20.0/24
dir fwd priority 375423 ptype main
tmpl src 10.12.12.10 dst 10.12.12.20
proto esp reqid 1 mode tunnel
src 192.168.10.0/24 dst 192.168.10.0/24
dir in priority 375423 ptype main
tmpl src 10.12.12.10 dst 10.12.12.20
proto esp reqid 1 mode tunnel
.....
```



Troubleshooting e Comandi Utili

- Monitor degli eventi

  `ip xfrm monitor`

  ..ad ogni pacchetto che passa per il tunnel se lasciate il monitor in esecuzione dovreste vedere un evento triggerarsi ad esempio:

```
Async event  (0x20)  timer expired
src 10.12.12.10 dst 10.12.12.20  reqid 0x1 protocol
esp  SPI 0xc0b3d0b8
```

Troubleshooting e Comandi Utili

- Regole iptables. Specifica per ipsec

```
ip route show table 220
192.168.10.0/24 via 10.12.12.10 dev
eth2 proto static src 192.168.20.254
```

mostra la route del tunnel

- Regole iptables. Route di default

```
ip route show table all
192.168.10.0/24 via 10.12.12.10 dev
eth2 table 220 proto static src
192.168.20.254
```

che mostra nella prima linea la regola più stringente per il tunnel ipsec

Check del tunnel

- Il primo check che possiamo fare è un ping tra i due server attraverso gli ip delle sottoreti figlie, ovvero:
  - Da R1 `ping 10.12.12.20`
  - Da R2 `ping 10.12.12.10`

- Il check fondamentale è il ping tra gli host distanti:
  - Da client `ping 192.168.20.20`
  - Da server `ping 192.168.10.10`

Check del tunnel - sniffing

- Installiamo tcpdump che è uno sniffer a riga di comando con **sudo apt install tcpdump**
- Su R1 lanciamo **tcpdump** da root e mettiamoci in ascolto sull'interfaccia che rappresenta il tunnel

  tcpdump -i eth2 -vnlp -A
- Su R2 o da server mandiamo un ping al client o a R1 usando come IP destinazione quello su LAN1
- Cosa notiamo sull'output?

Check del tunnel - sniffing

- Notiamo due pacchetti:
  - Un pacchetto ESP crittato sugli ip della rete tunnel
  - Il pacchetto del ping dal nodo R2 al R1
- Sembra quindi che il pacchetto sia “duplicato” una volta cifrato una seconda volta decrittato.
- In realtà quello che vediamo duplicato però è solo in un verso, non in entrambi come per i pacchetti ESP

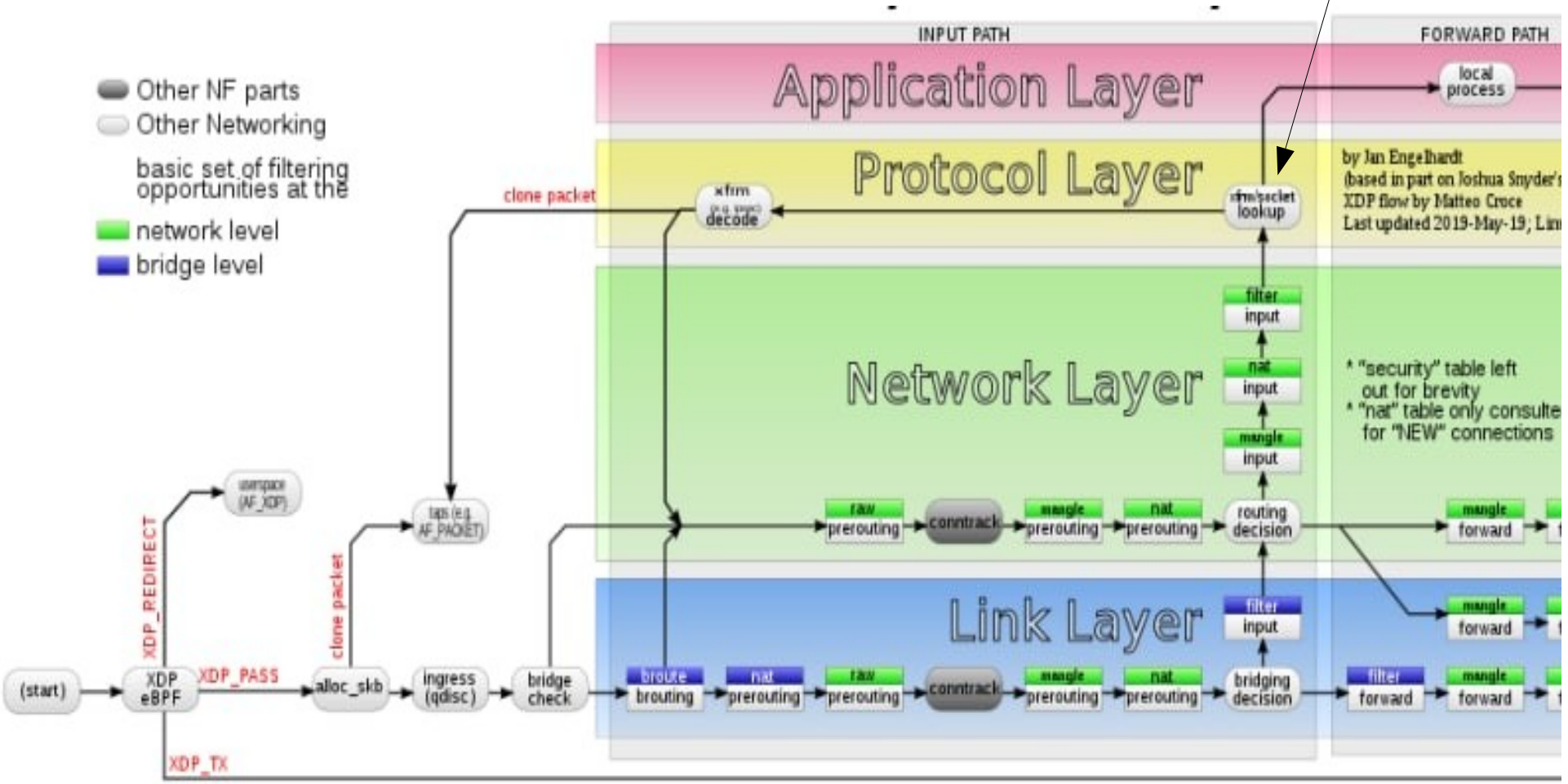
Check del tunnel - sniffing

- Quello che succede è dovuto al comportamento di ipsec e del flusso dei pacchetti in generale
- I pacchetti sono infatti cifrati direttamente dal kernel
- Il flusso dei pacchetti fa si che quanto arriva a livello protocol venga fatto il lookup xfrm dal kernel, il pacchetto viene quindi “duplicato” decrittato e rimesso in coda a livello network, per cui lo vediamo duplicato per questo motivo
- Nell’immagine successiva il riferimento sulla tabella di NetFilter

Check del tunnel - sniffing

Il primo pacchetto ESP viene clonato qui e reimesso in chiaro come pacchetto ICMP

Packet Flow in Netfilter



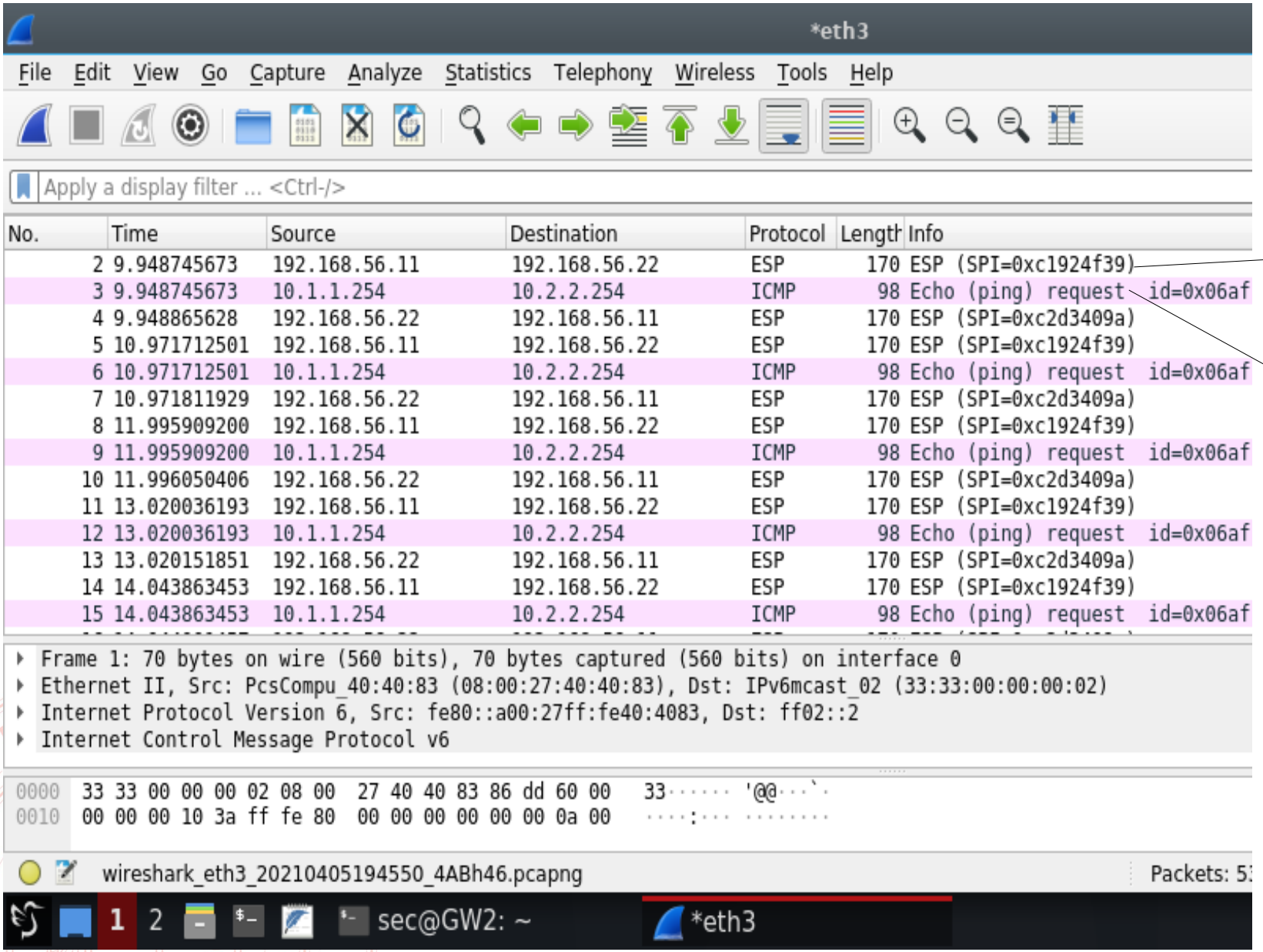
tcpdump --> wireshark

- Se vogliamo utilizzare wireshark per visualizzare più facilmente i dettagli, possiamo
  - registrare il traffico con tcpdump in un file pcap dalla VM

```
tcpdump -i eth2 -vnlp -A -w /vagrant/eth2.pcap
```

  - importarlo in wireshark lanciato sull'host
    - Menu file
    - Open
    - `<directory del Vagrantfile>/eth2.pcap`

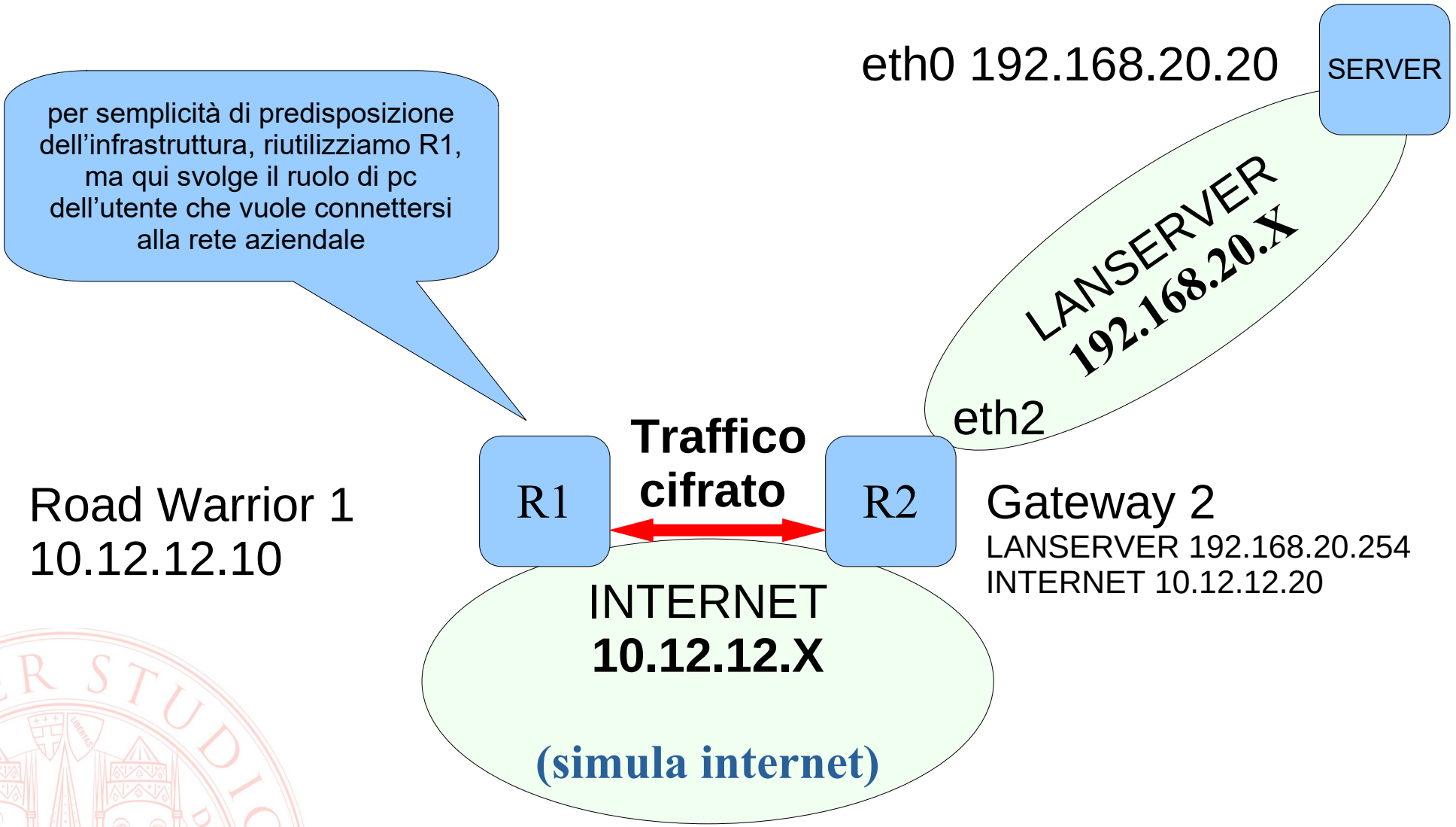
Check del tunnel wireshark



ESP packet

ICMP packet

Topologia per trasporto host-to-site



Indicazioni per esercizio h2s

Ipsec.conf di R1 Warrior

```
config setup

conn vpn1
authby=secret
auto=start
compress=no
type=tunnel
    left=%defaultroute
    right=10.12.12.20
    rightsubnet=192.168.20.0/24

include /var/lib/strongswan/ipsec.conf.inc
```

Ipsec.conf di R2

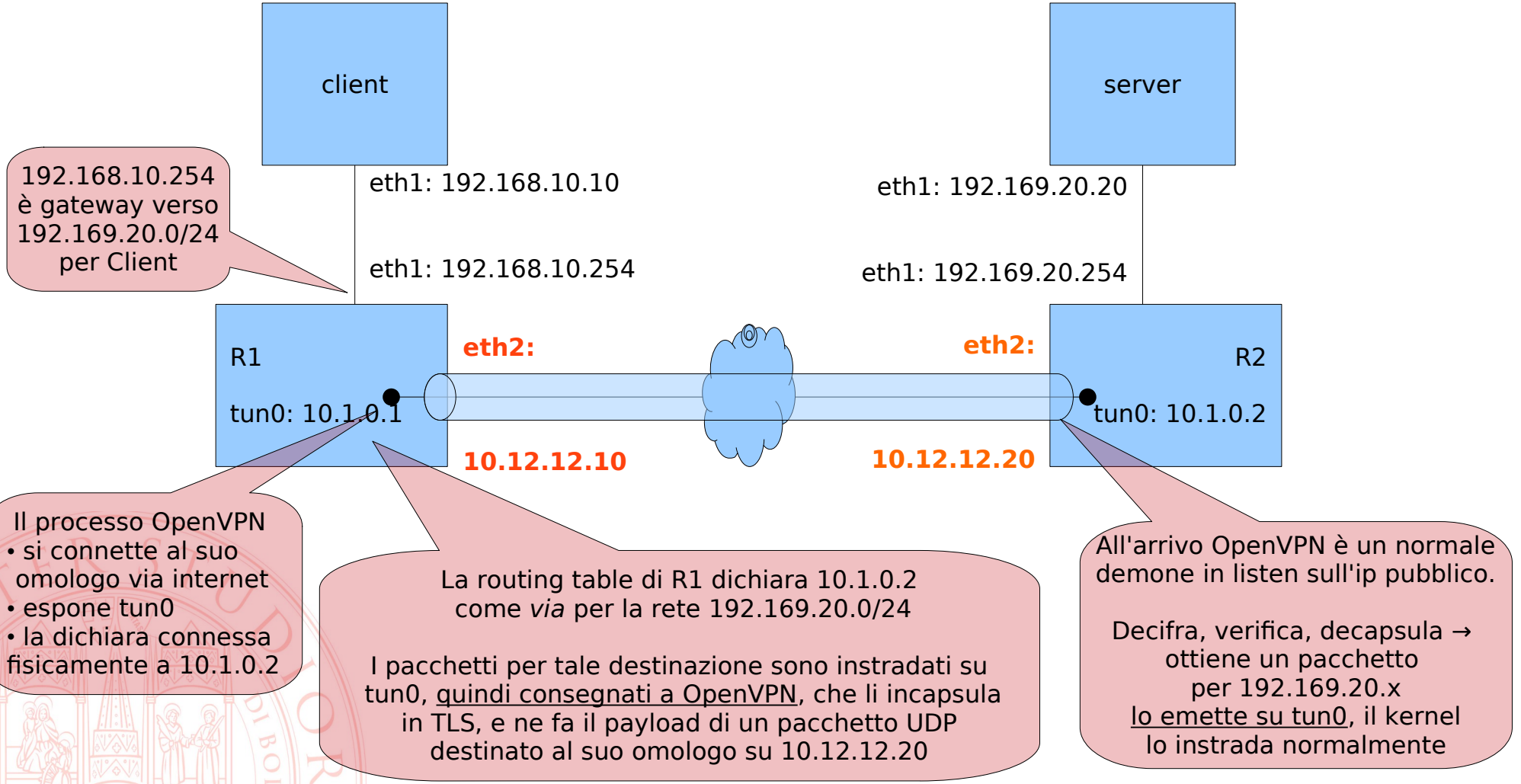
```
config setup

conn vpn1
authby=secret
auto=add
compress=no
type=tunnel
    left=%defaultroute
    leftsubnet=192.168.20.0/24
    right=%any

include /var/lib/strongswan/ipsec.conf.inc
```

OpenVPN

- **Per l'esercitazione, utilizzeremo la stessa topologia già impiegata con IPSec.**
  - **Eliminate la configurazione IPSec ripristinando le VM allo snapshot "rete ok" - mantenendo la "spunta" sulla finestra di conferma, potrete salvarla creando uno snapshot , chiamandolo "ipsec"**
- **OpenVPN riproduce con software in user space i concetti di transport e tunnel mode di IPSec**
  - *nota: deve girare come root per svolgere due operazioni privilegiate*:
    - la generazione di interfacce di rete virtuali, rispettivamente di tipo *tap* e *tun*
    - la scrittura di entry nelle tabelle di routing
  - fatto questo, qualunque applicazione può usare tali interfacce esattamente come quelle reali, con l'unica differenza (trasparente alle applicazioni) che:
    - i pacchetti inviati a un'interfaccia reale sono inviate al device driver della scheda hardware
    - i pacchetti inviati a un'interfaccia virtuale sono inviati al processo che le ha create

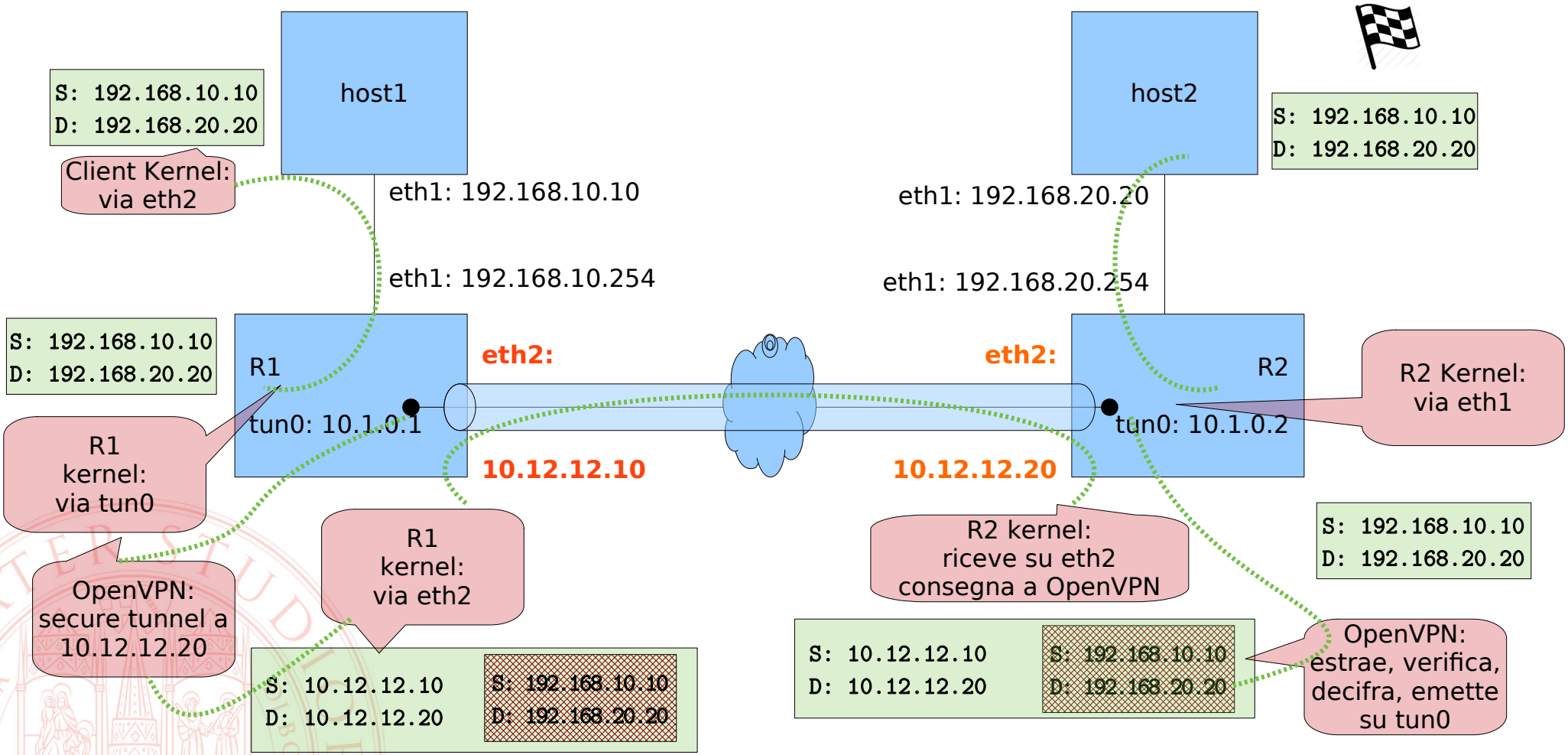
Tunnel mode

- Simuliamo una rete che collega due siti remoti:



Tunnel mode

- Le due reti vedono normale instradamento IP



Static key

La modalità “static key” di OpenVPN è la più semplice da abilitare:

- unica chiave di cifratura simmetrica condivisa fra Client e Server VPN
- pratica per configurazione statica tunnel mode
- non è possibile autenticare gli utenti

Configurazione con chiave condivisa

- Su ognuno dei GW usiamo i file dell cartella openvpn (sempre da utente root)

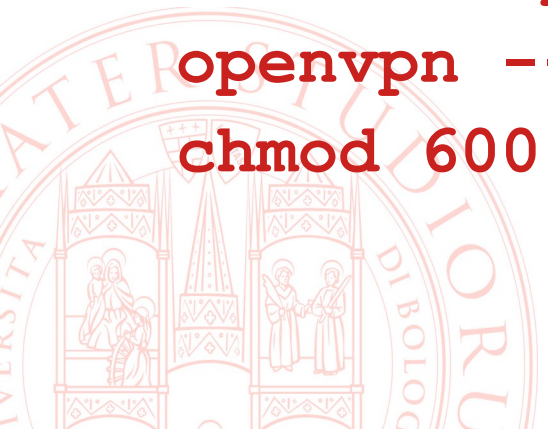
```
cd ~/vpn_files/openvpn
```

- Osserviamo la chiave condivisa

```
cat /etc/openvpn/static.key
```

per crearla sono stati usati i comandi

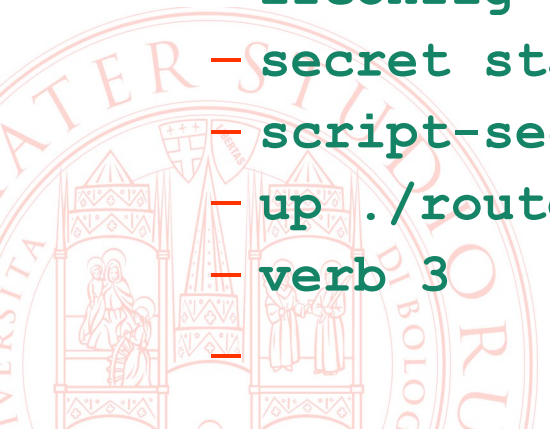
```
cd /etc/openvpn
openvpn --genkey --secret static.key
chmod 600 /etc/openvpn/static.key
```



Configurazione server OpenVPN

- Il software OpenVPN si comporta come una normale applicazione client-server. Un lato deve mettersi in ascolto, in attesa di connessioni, e l'altro deve connettersi per creare il tunnel.
- Arbitrariamente, utilizziamo R2 come server
  - confrontiamo la topologia che abbiamo illustrato col contenuto del file `/etc/openvpn/server.conf`

```
dev tun
local 10.12.12.20
ifconfig 10.1.0.2 10.1.0.1
secret static.key
script-security 3
up ./route.up
verb 3
```

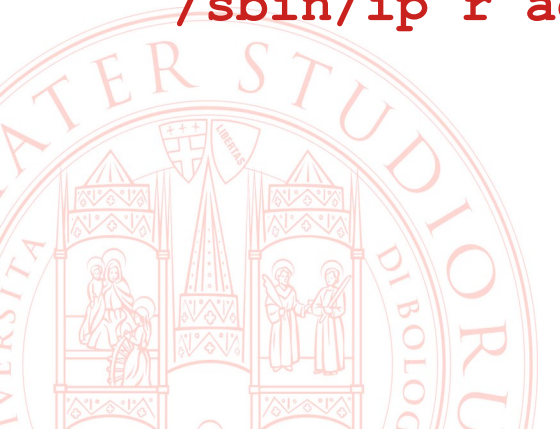


Configurazione rotta “2” → “1”

- OpenVPN in altri scenari ha varie direttive per impostare automaticamente le rotte, ma in questo esempio si usa una caratteristica più generale, che permette di eseguire uno script generico all’attivazione del tunnel
  - direttive `script-security 3` e `up ./route.up`

- il file `/etc/openvpn/route.up` di R2 contiene:

```
#!/bin/bash
/sbin/ip r del  192.168.10.0/24 2>/dev/null
/sbin/ip r add  192.168.10.0/24 via 10.1.0.1
```



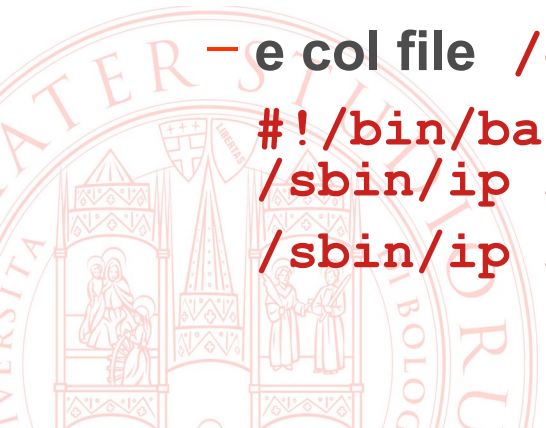
Configurazione client OpenVPN

- Corrispondentemente, utilizziamo R1 come client
  - confrontiamo la topologia che abbiamo illustrato col contenuto del file `/etc/openvpn/client.conf`

```
dev tun
remote 10.12.12.20
ifconfig 10.1.0.1 10.1.0.2
secret static.key
script-security 3
up ./route.up
verb 3
```

  - e col file `/etc/openvpn/route.up`:

```
#!/bin/bash
/sbin/ip r del  192.168.20.0/24 2>/dev/null
/sbin/ip r add  192.168.20.0/24 via 10.1.0.2
```



Avvio e test

- Avviamo il servizio su entrambi i gateway con

  `openvpn --config <file_conf>`

  - Nota: non riparte automaticamente al boot a meno che non si dia anche il comando

- Test vari:
  - ping
  - traceroute
  - tcpdump sulle diverse interfacce (reali e virtuali)

Road Warrior

- Viene così definita la configurazione di un client su rete pubblica che vuole accedere alla rete aziendale
- Peculiarità:
  - identità di rete molto variabile
    - meglio ricorrere all'identificazione dell'utente
  - non necessariamente la rete locale è una rete fidata
    - non si vuole una configurazione site-to-site
    - non si vuole propagare alla rete locale il trasporto di servizi rischiosi, come il discovery automatico di risorse
  - potrebbe essere utile vedere il RW come parte della LAN aziendale
    - bisogna trasportare anche il layer 2

Road Warrior bridged vs. routed

Per consentire la comunicazione tra il Client VPN e gli host della rete remota vi sono due possibili strade:

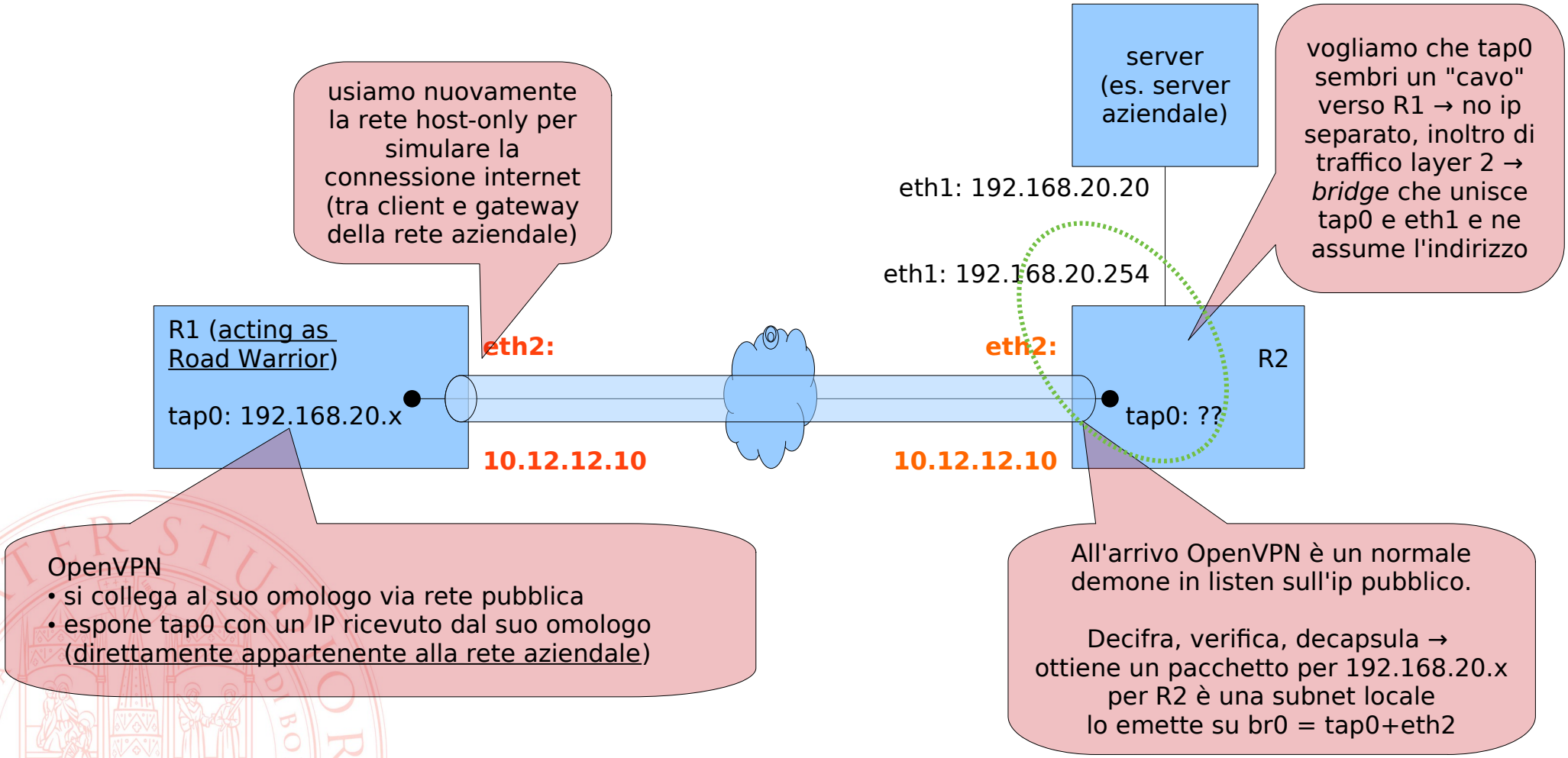
- configurare la tabella di routing del Server VPN per instradare i pacchetti da e verso la rete del Client
- configurare un bridge ethernet per connettere l'interfaccia VPN del Server con l'interfaccia ethernet connessa alla rete locale
  - questa soluzione consente al client l'uso di protocolli basati su LAN broadcast (discovery di servizi ed enumerazione di risorse)
  - l'assegnamento di un ip della rete aziendale semplifica la configurazione di servizi e firewall
- Nel seguito verrà descritto come configurare una connessione VPN utilizzando la modalità SSL/TSL e il bridging delle interfacce

TUN vs. TAP

- L'interfaccia tun vista in precedenza è un puro artificio per creare una connessione punto-punto tra i due gateway mediata da OpenVPN
- Dal punto di vista delle applicazioni, gli indirizzi delle interfacce tun sono trasparenti e non appartengono a nessuna delle subnet effettivamente utilizzate da host1 e host2, servono solo per il *routing*
- Per rendere una macchina remota virtualmente parte di una rete locale si ricorre al modo TAP, tipicamente associato al *bridging*
- NON è un transport mode come quello di IPSec, perché genera un'interfaccia con un proprio IP e comunque incapsula i pacchetti da e per tale indirizzo in un tunnel fatto sugli indirizzi pubblici
- La differenza con TUN è che incapsula anche il Layer 2

TAP mode

- **Simuliamo una rete che collega un host a una rete remota come se ne facesse fisicamente parte**



TAP mode

- Simuliamo una rete che collega un host a una rete remota come se ne facesse fisicamente parte